УДК 519.682.1

СПОСОБ ПРЕОБРАЗОВАНИЯ КС-ГРАММАТИКИ В *LL*(1)-ГРАММАТИКУ

Ю. Д. Рязанов

Белгородский государственный технологический университет им. В.Г. Шухова

Поступила в редакцию 03.04.2014 г.

Аннотация. В статье рассматриваются КС-грамматики без левой рекурсии, не принадлежащие классу *LL*(1)-грамматик только из-за наличия конфликта между такими парами правил, в которых одно правило имеет пустую правую часть. Предлагается способ устранения конфликтов и алгоритм, который либо преобразует КС-грамматику в *LL*(1)-грамматику, либо сообщает о невозможности преобразования этим способом. Предложенный способ расширяет класс КС-грамматик, которые можно преобразовать в *LL*(1)-грамматику.
Ключевые слова: КС-грамматики, *LL*(1)-грамматики, *LL*(1)-язык, множество выбора, преобразование грамматик.
Annotation. Context-free grammars without left-recursion, which are not *LL*(1) grammars is being considered in this article. That grammars are not *LL*(1) grammars, because they have conflict between such pairs of rules, which have one rule with empty right part. Author suggest method for solving conflicts and algorithm, which transformed grammar into *LL*(1) grammar or send message of invalid operation. This method expanded the class of context-free grammars, which can be transformed to *LL*(1) grammar.
Keywords: context-free grammars, *LL*(1) grammars, *LL*(1) language, set of choice, transforming grammars.

При построении нисходящих обработчиков языков используются *LL*-грамматики [1]. Наиболее эффективными получаются обработчики, построенные по *LL*(1)-грамматике. Исходная КС-грамматика языка может не принадлежать классу *LL*(1)-грамматик даже если язык принадлежит классу *LL*(1)-языков, т. е. если он порождается какой-либо *LL*(1)-грамматикой. В этом случае целесообразно преобразовать исходную грамматику в *LL*(1)-грамматику. В работе [1] показано, что проблема распознавания, существует ли для данной КС-грамматики, не являющейся *LL*(1)-грамматикой, эквивалентная ей *LL*(1)-грамматика, не разрешима. Следовательно, не существует алгоритма, который бы определял, порождает ли заданная КС-грамматика *LL*(1)-язык и, если так, преобразовывал бы ее в *LL*(1)-грамматику. Однако для некоторых КС-грамматик такое преобразование возможно. Традиционный способ получения *LL*(1)-грамматики, рекомендуемый практически во всех работах, в которых рассматриваются вопросы построения нисходящих обработчиков языков, в том числе в [1–6], основан на устранении левой рекурсии, выполнении факторизации и замены края. В результате таких преобразований может быть получена *LL*(1)-грамматика, произойти зацикливание [3], либо может быть получена не *LL*(1)-грамматика с ε-правилами, в которой правые части любых двух не ε-правил с одинаковыми левыми частями начинаются различными терминалами (в статье такая грамматика названа грамматикой особого вида). В статье предлагается способ преобразования грамматики особого вида в *LL*(1)-грамматику. Это позволит расширить класс КС-грамматик, которые можно преобразовать в *LL*(1)-грамматику, по сравнению со способом, использующим устранение левой рекурсии, факторизации и замены края.

К классу *LL*(1)-грамматик относятся КС-грамматики, у которых множества выбо-

ра любых двух правил с одинаковыми левыми частями не пересекаются, т. е. если в грамматике есть правила $A \to \alpha_1$ и $A \to \alpha_2$, то $ВЫБОР(A \to \alpha_1) \cap ВЫБОР(A \to \alpha_2) = \varnothing$. Для нахождения множества выбора правила $A \to \alpha$ используются множество первых цепочки α $ПЕРВ(\alpha)$ и множество следующих за нетерминалом A $СЛЕД(A)$. Множество $ПЕРВ(\alpha) = \{x \mid \exists \alpha \Rightarrow * x\beta\}$, т. е. терминал, который может быть первым в какой либо цепочке, выводимой из цепочки α, принадлежит множеству $ПЕРВ(\alpha)$. Если S – начальный нетерминал, то множество $СЛЕД(A) = \{x \mid \exists S \Rightarrow * \beta A x y\}$, т. е. терминал (или концевой маркер), который может располагаться непосредственно после нетерминала A в какой либо цепочке вывода, принадлежит множеству $СЛЕД(A)$. Множество выбора правила $A \to \alpha$ определяется следующим образом: если из α нельзя вывести пустую цепочку ε, то $ВЫБОР(A \to \alpha) = ПЕРВ(\alpha)$, если же из α можно вывести пустую цепочку ε, то $ВЫБОР(A \to \alpha) = ПЕРВ(\alpha) \cup СЛЕД(A)$. Алгоритмы нахождения множеств $ПЕРВ(\alpha)$, $СЛЕД(A)$ и множеств выбора известны [1, 4].

Если исходная грамматика не является *LL*(1)-грамматикой, то, как было сказано в начале статьи, рекомендуется устранить левую рекурсию и, если в грамматике будут правила $A \to \alpha_1$ и $A \to \alpha_2$, такие, что $\alpha_1 \neq \varepsilon$, $\alpha_2 \neq \varepsilon$ и $ВЫБОР(A \to \alpha_1) \cap ВЫБОР(A \to \alpha_2) \neq \varnothing$, выполнить замену края и левую факторизацию для правил с нетерминалом A в левой части. В результате выполнения таких преобразований появляются новые нетерминалы и новые правила, для которых, возможно, потребуется снова выполнить эти преобразования. Процесс может «зациклиться» и *LL*(1)-грамматику с конечным множеством правил нельзя будет получить. Такое зацикливание может быть обнаружено в процессе выполнения преобразований.

Допустим, в процессе выполнения описанных выше преобразований, получена грамматика, в которой для любых двух правил $A \to \alpha_1$ и $A \to \alpha_2$, таких, что $\alpha_1 \neq \varepsilon$, $\alpha_2 \neq \varepsilon$ и множества выбора не пересекаются $(ВЫБОР(A \to \alpha_1) \cap ВЫБОР(A \to \alpha_2) = \varnothing)$. Выполняя замену края, такую грамматику можно преобразовать в грамматику *особого вида*, в которой правая часть каждого правила либо начинается с терминала, либо представляет собой пустую цепочку (ε-правило), причем правые части любых двух не ε-правил с одинаковой левой частью начинаются различными терминалами. Грамматика особого вида может не быть *LL*(1)-грамматикой, если в ней есть хотя бы два правила $A \to \alpha$ и $A \to \varepsilon$ $(\alpha \neq \varepsilon)$, таких, что их множества выбора пересекаются. Если, в этом случае, $x \in ВЫБОР(A \to \alpha) \cap ВЫБОР(A \to \varepsilon)$, то будем говорить, что имеет место *конфликт по терминалу x при конфликтующих правилах* $A \to \alpha$ и $A \to \varepsilon$. В *LL*(1)-грамматике таких конфликтов и конфликтующих правил нет.

В статье предлагается способ, который либо преобразует грамматику особого вида в *LL*(1)-грамматику, либо сообщает о невозможности получения *LL*(1)-грамматики этим способом.

Целью преобразования является устранение конфликтов по всем терминалам. Суть предлагаемого способа получения *LL*(1)-грамматики заключается в следующем.

1. Определить множество M терминалов, по которым имеет место конфликт. Если $M = \varnothing$, то *LL*(1)-грамматика получена.

2. Последовательно выполнять попытки исключения конфликтов по терминалам множества M. Если $x \in M$ и попытка исключения конфликта по терминалу x уже выполнялась, то попытку считать неудачной. Если все попытки удачные, выполнить п. 1, иначе получить *LL*(1)-грамматику данным способом нельзя.

То, что количество попыток исключения конфликта по любому терминалу не более одной (см. п. 2), гарантирует завершение алгоритма через конечное число шагов (не более количества терминалов в грамматике).

Для выполнения преобразования введем множество M^A для нетерминала A, которое определяется следующим образом:

1) $M^A = \{A\}$;

2) если в грамматике есть правило $X \to \alpha B \beta$, $\beta \Rightarrow^* \varepsilon$ и $B \in M^A$, то $M^A := M^A \cup \{X\}$, т. е. нетерминал X добавить в множество M^A;

3) повторять п. 2, пока множество M^A растет.

Для $X \in M^A$ верно, что

$$СЛЕД(X) \subseteq СЛЕД(A).$$

Попытка исключения конфликта по терминалу x при конфликтующих правилах $A \to \alpha$ и $A \to \varepsilon$ заключается в следующем.

Сформировать множество M^A. Так как $ВЫБОР(A \to \varepsilon) = СЛЕД(A)$, то при наличии конфликта по терминалу x хотя бы в одном правиле грамматики должна быть подцепочка $B\beta$, где $B \in M^A$ и $x \in ПЕРВ(\beta)$. Будем обрабатывать ситуации, в которых $\beta = xy$, иначе попытку считаем неудачной.

Каждую подцепочку вида Bx в правилах грамматики заменить новым нетерминалом N и ввести правило $N \to Bx$. Полученную грамматику преобразовать в грамматику особого вида, выполняя замену края и левую факторизацию. Если в процессе преобразования в правилах появятся подцепочки вида Bx, то заменить их нетерминалом N. Если грамматику особого вида получить не удалось, попытку исключения конфликта считать неудачной. При выполнении преобразований следует учитывать то, что после замены края некоторые нетерминалы грамматики могут стать недостижимыми и правила, их содержащие, нужно удалить из грамматики.

Пример 1.

Грамматика особого вида задана множеством правил:

1. $S \to aBa$
2. $S \to cbAaSc$
3. $S \to \varepsilon$
4. $A \to abA$
5. $A \to \varepsilon$
6. $B \to aA$
7. $B \to \varepsilon$

Определим множества выбора ε-правил:

$$ВЫБР(S \to \varepsilon) = \{c, -|\}$$

$$ВЫБР(A \to \varepsilon) = \{a\} \quad ВЫБР(B \to \varepsilon) = \{a\}$$

В этой грамматике имеет место конфликт по терминалу с в правилах 2, 3 и по терминалу a в правилах 4, 5 и 6, 7.

Для устранения конфликта по терминалу с определим множество $M^S = \{S\}$, подцепочку Sc во втором правиле заменим на нетерминал N_1 и введем правило $N_1 \to Sc$.

Для устранения конфликта по терминалу a в правилах 4 и 5 определим множество $M^A = \{A, B\}$, подцепочку Aa во втором правиле заменим нетерминалом N_2 и введем правило $N_2 \to Aa$, а подцепочку Ba в первом правиле заменим нетерминалом N_3 и введем правило $N_3 \to Ba$.

Для устранения конфликта по терминалу a в правилах 4 и 5 определим множество $M^B = \{B\}$. Подцепочка Ba в первом правиле уже заменена нетерминалом N_3 и введено правило $N_3 \to Ba$.

В результате получим грамматику:

$S \to aN_3$ $A \to abA$ $B \to aA$ $N_1 \to Sc$

$S \to cbN_2N_1A \to \varepsilon$ $B \to \varepsilon$ $N_2 \to Aa$

$S \to \varepsilon$ $N_3 \to Ba$

Теперь преобразуем правила для нетерминалов N_1, N_2, N_3. Будем выполнять замену края, и, если появятся подцепочки Sc, Aa, или Ba, заменим их на N_1, N_2 или N_3 соответственно. Правила для нетерминалов S, A и B не изменятся, поэтому их переписывать не будем. Правила для N_1, N_2 и N_3 примут вид:

$N_1 \to aN_3c$ $N_2 \to abN_2$ $N_3 \to aN_2$

$N_1 \to cbN_2N_1c$ $N_2 \to a$ $N_3 \to a$

$N_1 \to c$

Заметим, что нетерминалы A и B стали недостижимыми и правила, их содержащие, исключим из грамматики. Для получения грамматики особого вида выполним левую факторизацию:

$S \to aN_3$ $N_1 \to aN_3c$ $N_2 \to aN_5$ $N_3 \to aN_6$

$S \to cbN_2N_1$ $N_1 \to cN_4$ $N_5 \to bN_2$ $N_6 \to N_2$

$S \to \varepsilon$ $N_4 \to bN_2N_1cN_5 \to \varepsilon$ $N_6 \to \varepsilon$

$N_4 \to \varepsilon$

и замену края в правиле $N_6 \to N_2$:

$S \to aN_3$ $N_1 \to aN_3c$ $N_2 \to aN_5$ $N_3 \to aN_6$

$S \to cbN_2N_1$ $N_1 \to cN_4$ $N_5 \to bN_2$ $N_6 \to aN_5$

$S \to \varepsilon$ $N_4 \to bN_2N_1cN_5 \to \varepsilon$ $N_6 \to \varepsilon$

$N_4 \to \varepsilon$

Определим множества выбора для ε-правил:

$$ВЫБОР(S \to \varepsilon) = \{\text{-}|\},$$

$$ВЫБОР(N_5 \to \varepsilon) = \{a, c, \text{-}|\},$$

$$ВЫБОР(N_4 \to \varepsilon) = ВЫБОР(N_6 \to \varepsilon) = \{c, \text{-}|\}.$$

Видим, что конфликты устранены, $LL(1)$-грамматика получена.

Пример 2.

Грамматика особого вида задана множеством правил:

1. $S \to cSab$
2. $S \to bA$
3. $A \to aS$
4. $A \to \varepsilon$

Определим множества выбора ε-правила: $ВЫБР(A \to \varepsilon) = \{a, \text{-}|\}$.

В этой грамматике имеет место конфликт терминалу a в правилах 3, 4.

Для устранения конфликта определим множество $M^A = \{A, S\}$, подцепочку Sa в первом правиле заменим нетерминалом N_1 и введем правило $N_1 \to Sa$. В результате получим грамматику:

$S \to cN_1b$ $\quad A \to aS$ $\quad N_1 \to Sa$

$S \to bA$ $\quad A \to \varepsilon$

Выполним замену края в правиле для нетерминала N_1:

$S \to cN_1b$ $\quad A \to aS$ $\quad N_1 \to cN_1ba$

$S \to bA$ $\quad A \to \varepsilon$ $\quad N_1 \to bAa$

Определим множества выбора ε-правила: $ВЫБР(A \to \varepsilon) = \{a, \text{-}|\}$.

После выполнения преобразований опять возник конфликт по терминалу a. Поэтому попытку устранить конфликт считаем неудачной и $LL(1)$-грамматику предложенным способом получить нельзя.

Итак, предложен способ преобразования КС-грамматики в $LL(1)$-грамматику, представлены примеры его применения. Этот способ может быть улучшен, если пытаться устранить конфликт в случаях, в которых конфликт по терминалу x при конфликтующих правилах $A \to \alpha$ и $A \to \varepsilon$ возникает из-за того, что в грамматике есть правило, содержащее подцепочку $B\beta$, где $B \in M^A$, $x \in ПЕРВ(\beta)$ и первый символ в цепочке β – нетерминал. Для этого нужно выполнить замену первого нетерминала в цепочке β и затем применить предложенный способ устранения конфликта. Другое улучшение предложенного способа заключается в поиске условия, при котором попытку устранения конфликта нужно считать неудачной. Улучшения способа должны быть направлены на увеличение мощности множества КС-грамматик, порождающих $LL(1)$-языки, которые могут быть преобразованы в $LL(1)$-грамматики.

СПИСОК ЛИТЕРАТУРЫ

1. *Ахо А., Ульман Дж.* Теория синтаксического анализа, перевода и компиляции. – М. : Мир, 1978.

2. *Ахо А., Сети Р., Лам С.М., Ульман Дж.* Компиляторы: принципы, технологиии и инструменты. М. : Вильямс, 2008.

3. *Кревский И.Г., Селиверстов М.Н., Григорьева К.В.* Формальные языки, грамматики и основы построения трансляторов: Учебное пособие / Под ред. А.М. Бершадского. – Пенза : Пенз. гос. ун-т, 2002.

4. *Льюис Ф., Розенкранц Д., Стирнз Р.* Теоретические основы проектирования компиляторов. – М. : Мир, 1979.

5. *Серебряков В.А.* Теория и реализация языков программирования. М. : Физматлит, 2012.

6. *Фомичев В.С.* Формальные языки, грамматики и автоматы: Курс лекций. URL: http://www.eltech.ru/misc/LGA_2007_FINAL/Index.html (дата обращения 03.04.2014).

Рязанов Юрий Дмитриевич – Белгородский государственный технологический университет им. В.Г. Шухова, доцент кафедры программного обеспечения вычислительной техники и автоматизированных систем.
Тел.: +7 (910) 325-73-75.
E-mail: Ryazanov.iurij@yandex.ru

Ryazanov Yury Dmitrievich – BSTU after V.G. Shukhov, PhD in engineering, assistant professor of the department of software computer and automated systems.
Phone: +7 (910) 325-73-75.
E-mail: Ryazanov.iurij@yandex.ru